(12) UK Patent Application (19) GB (11) 2 357 596 (13) A

(43) Date of A Publication 27.06.2001

(21) Application No 9930070.9

(22) Date of Filing 20.12.1999

(71) Applicant(s)
University College London
(Incorporated in the United Kingdom)
Gower Street, LONDON, WC1E 6BT, United Kingdom

(72) Inventor(s)
Mark Levene
Nadav Zin

(74) Agent and/or Address for Service
Carpmaels & Ransford
43 Bloomsbury Square, LONDON, WC1A 2RA,
United Kingdom

(51) INT CL[7]
G06F 17/30

(52) UK CL (Edition S)
G4A AUDB

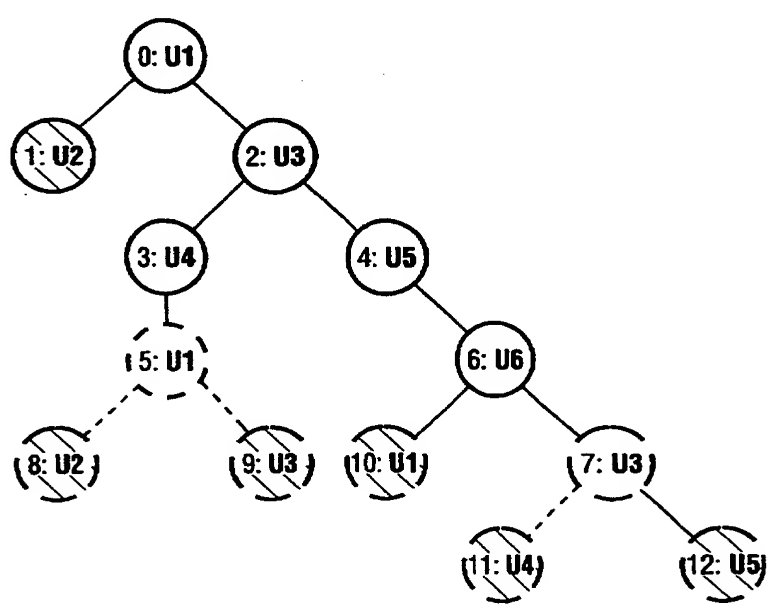
(56) Documents Cited
EP 0981097 A1
"Toward User-Centric Navigation...", 25 Apr 1997, at www.scope.gmd.de/info/www6/posters/744/ucn.htm

(58) Field of Search
UK CL (Edition R) G4A AUDB
INT CL[7] G06F 17/30
Online: COMPUTER, EPODOC, INSPEC, Internet, JAPIO, TDB, WPI

(54) Abstract Title
A navigation engine for assessing the quality of a trail between linked pages

(57) A navigation engine for finding the best trails in the World-Wide-Web or any other hypertext system with respect to a user query, given one or more starting URLs U1 each uniquely identifying e.g. a Web page. The navigation engine builds a navigation tree, which simulates user navigation, by following links with probability proportional to the score of the trail, induced by the destination URL of the link followed, with respect to the query. A tip node in the navigation tree corresponds to a URL which can be browsed by traversing an out-link from the Web page associated with the URL, or to a node in the navigation tree whose URL is associated with a Web page having no out-links. The best trail for a given starting URL and an input query is the highest ranking trail induced by the tip nodes of the final state of the navigation tree. The navigation engine utilises two stages: an exploration stage and a convergence stage, each comprising a fixed number of iterations. The best trail navigation engine can be used as a support tool for browsing or as a plug-in to a search engine for the purpose of assisting the user during navigation.

FIG. 3 An example navigation tree



At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

GB 2 357 596 A

FIG. 1

Flowchart of the best trail algorithm

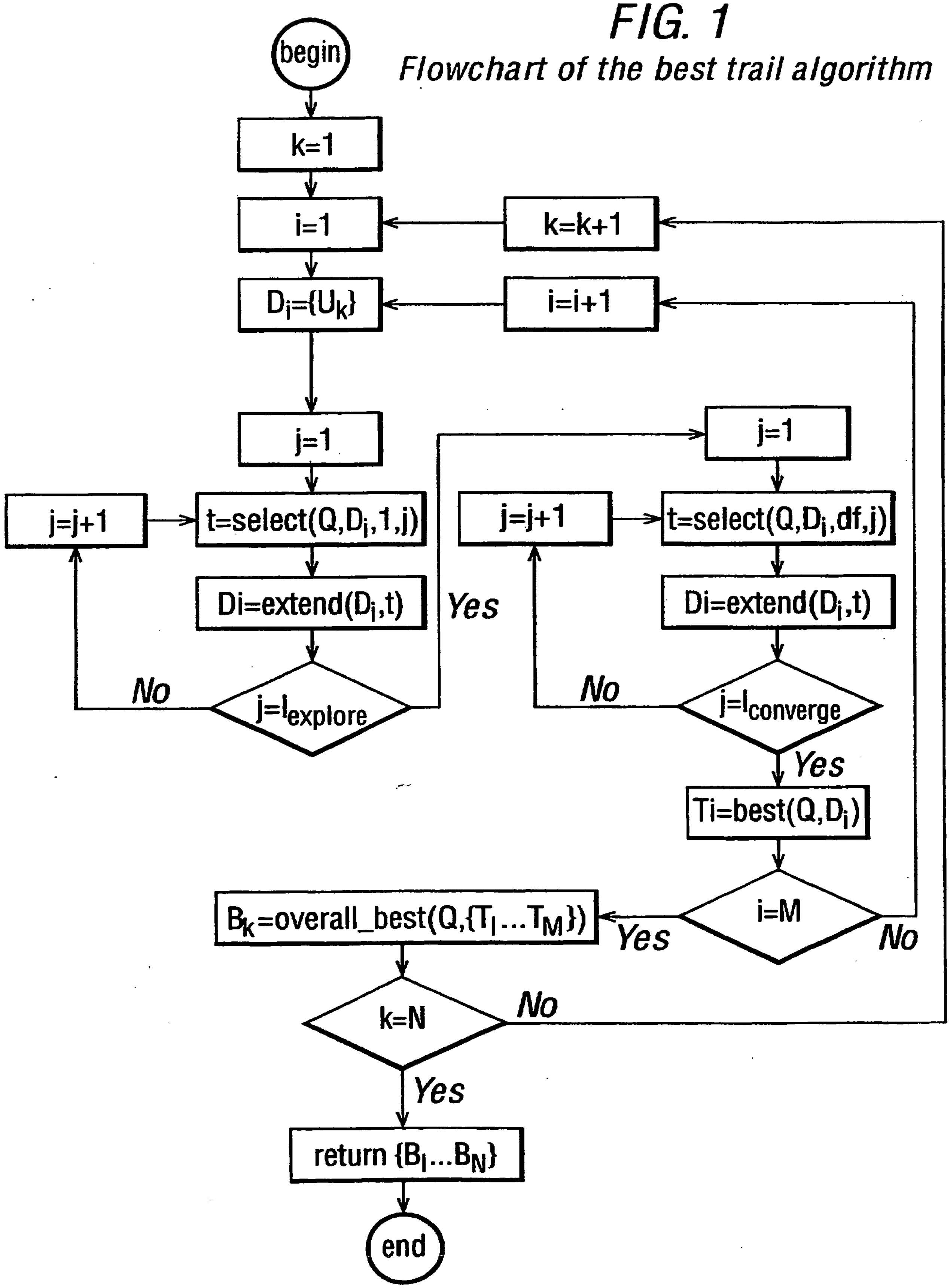


FIG. 2 An example Web topology

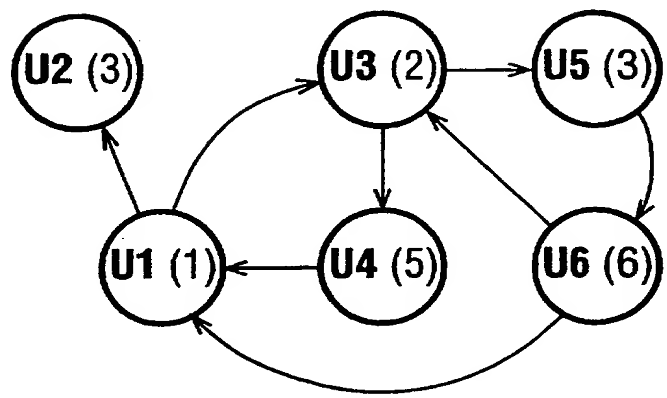


FIG. 3 An example navigation tree

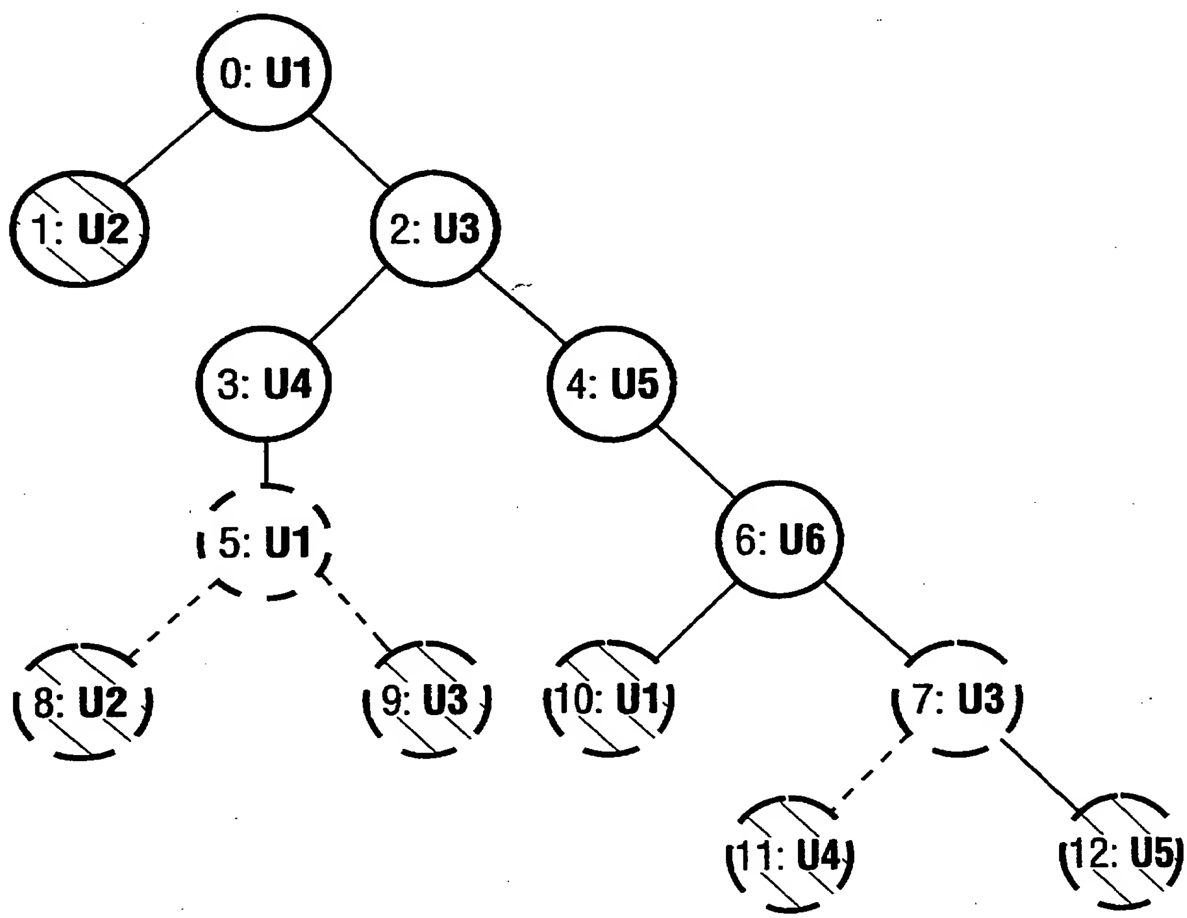
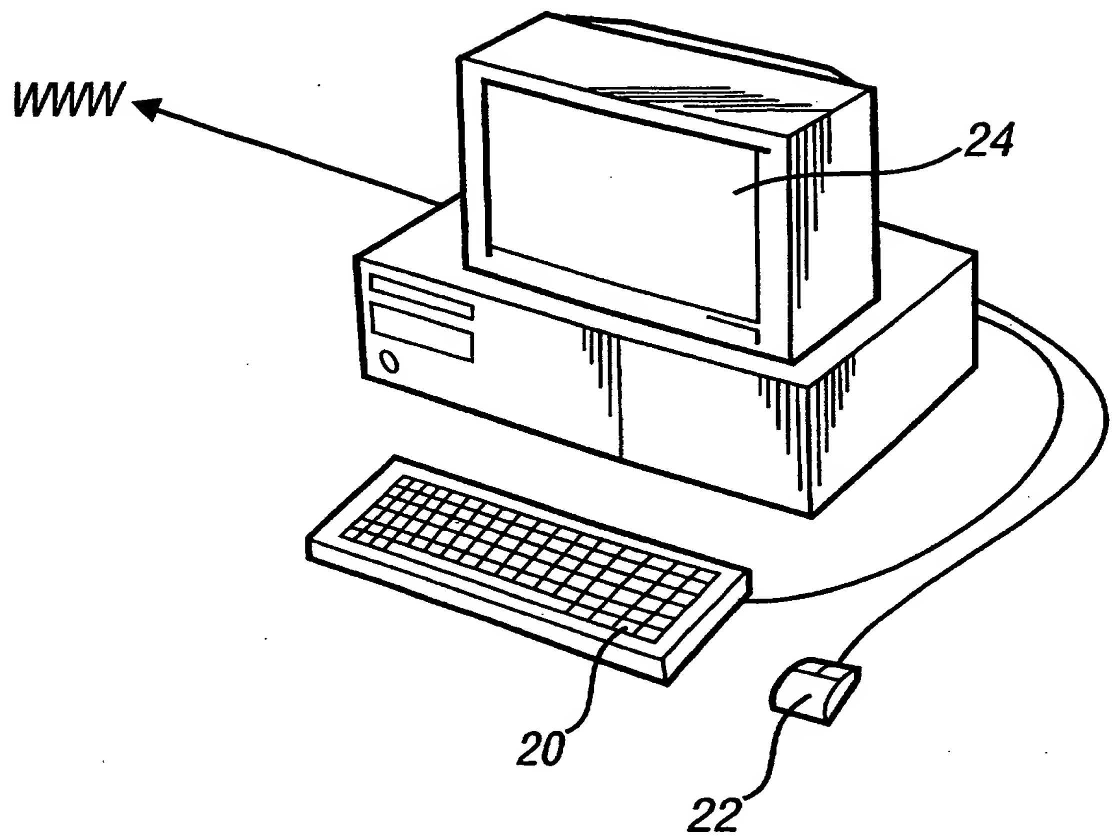


FIG. 4



2357596

A NAVIGATION ENGINE FOR ASSESSING THE QUALITY OF A TRAIL BETWEEN PAGES IN A NETWORK

The present invention relates to the field of browsing and navigation for the purpose of finding preferred trails between pages in a network, and more particularly to a navigation engine and associated method which assesses trails between pages in the World-Wide-Web or in any other hypertext system to assist in finding relevant information.

The environment in which the present invention operates is the World-Wide-Web (known as the Web) or any other hypertext system. The Web can be viewed as a hypertext database containing nodes, which are the Web pages, and links between these nodes defining its topology. Each Web page has a unique identifier describing where the page resides and how to retrieve it. The mechanism used is that of a *Unified Resource Locator*, or simply URL, which specifies the unique path for locating the Web page. Every link connects two nodes. The node we start at is called the *anchor* node and the node we finish at is called the *destination* node.

The process of *navigation* (colloquially known as "surfing") is that of following links and inspecting (or browsing) the contents of Web pages visited during this process. A navigation session results in the user visiting a sequence of Web pages, which is called a *trail*. A trail is represented by the sequence of URLs associated with its pages. For example, a user's trail may be the sequence of URLs:

$$U_1, U_2, U_3, U_2, U_1, U_4.$$

During the navigation process users may become "lost in hyperspace", meaning that they become disoriented in terms of what to do next and how to return to a previously browsed Web page. In this situation users may lose the context in which they are browsing and need assistance in finding their way. This problem is known as the *navigation problem*.

Even if a user does not become lost in hyperspace, many of the links suggested by a particular Web page will not provide useful information to the user. This is because the links highlighted on a Web page are set up by the Web page provider and are not specifically related in any way to a particular query being pursued by the user. Hence, much time can be wasted by accessing irrelevant Web pages in this fashion. Thus, it would be extremely beneficial for a user to know which of the linked Web pages are likely to be relevant to a particular query, and hence which should be accessed by the user. The present invention is aimed at providing this facility, and more particularly to the provision of a system which will assess possible Web pages in a trail to help produce a trail of relevant Web pages which can be followed by a user.

In accordance with the foregoing, the present invention provides a navigation engine which uses a query defining a subject of interest to a user to select links between relevant pages in a network of linked textual or multi-media information, the navigation engine being able to assess the suitability of a plurality of links forming a trail based on the relevance of the pages in the trail, wherein the navigation engine provides an output related to the suitability to a user of various trails assessed.

The output preferably includes a list of suitable trails available to be accessed by a user. The list of suitable trails may be in order of suitability, with the most suitable trail listed first.

The relevance of a page in a network is preferably assessed based on the relevance of the page with respect to a query.

In a particular embodiment, a score is allocated to indicate the relevance of a page with respect to a query.

The suitability of a trail may be calculated based on a chosen scoring function. For example, the scoring function preferably involves at least one of the following:

(a) the average score for a page in the trail with respect to a query, taking into account each step in the trail;

(b) the average score of a page in the trail with respect to a query, counting each page only once even if it appears in the trail more than once;

(c) the sum of the scores of pages in the trail with respect to a query, counting each distinct page only once even it appears more than once in the trail, divided by the total number of pages in the trail irrespective of whether a page appears more than once in the trail; and

(d) the sum of discounted scores of the pages in the trail with respect to a query, were the discounted score of U_i, the page in the ith position in the trail, is the score of U_i with respect to the query multiplied by γ raised to the power of $(i - 1)$ where γ is a real number strictly between zero and one, i.e. the discounted score of a trail, $U_1, U_2, \ldots, U_m$, is equal to $\sum_{i=1}^{m} s_i \cdot \gamma^{i-1}$, where s_i is the score of U_i with respect to the query.

The trails are preferably ordered based on the result of the scoring function.

In a preferred embodiment, the trails are ranked by score, the highest score reflecting the best trail.

When assessing a trail, the trail may end with a page having no out-link.

Assessment of a plurality of trails preferably comprises an exploration stage and a convergence stage.

The exploration stage preferably includes extending trail lengths and scoring the trails that are induced.

The convergence stage preferably assesses which induced trails are more suitable and gives these trails more weight at each iteration based on their ranking.

Preferably an assessment of trails is conducted over sufficient iterations to produce a useful output. Any number of iterations may be applied, as appropriate.

Although not limited to such, the network is preferably a hypertext system, such as the World-Wide-Web.

As will be appreciated, a navigation engine according to the present invention is ideally suited for use when loaded into a computer system for connection to a network. Hence, the navigation engine may be accessible through the Web. Alternatively, it may be supplied to a user in the form of computer software stored on a carrier, such as a compact disk or floppy disk.

The present invention further provides a system for facilitating exploration by a user of a network of linked textual or multi-media information, the system comprising:

a user interface for receiving a query which defines a subject of interest to the user; and

a navigation engine as described or claimed herein.

A specific embodiment of the present invention is now described, by way of example only, with reference to the accompanying drawings, in which:-

Figure 1 is a flow chart depicting an embodiment of the algorithm for producing a best trail according to the present invention;

Figure 2 is a schematic representation of pages in a network (or Web); and

Figure 3 shows an example of a navigation tree, which could result from the Web topology shown in Figure 2; and

Figure 4 shows a user interface workstation for using a navigation engine according to the present invention to surf a network such as the World-Wide Web.

The context of a navigation session is a query, which normally would be a set of keywords. The query can be viewed as the goal of the navigation session in the sense that the user would like to follow a trail which maximises the suitability of the trail to the query. The suitability of a trail to the query is realised by its *score*,

which is a function of the scores of the individual Web pages of the trail with respect to the query. (We assume that scores of trails are numeric and trails having higher scores are more relevant to the query.) The score of a page with respect to a query indicates how closely the page contents match the query, i.e. how relevant the Web page is to the query. (We assume that scores of Web pages are numeric and Web pages having higher scores are more relevant to the query.) The scoring of individual Web pages with respect to a query is realised by information retrieval techniques. An example of a scoring method for a trail with respect to a query is that of taking the average score of its pages with respect to the query; other alternative trail scoring methods are described later.

A navigation engine according to the present invention, which uses a *best trail algorithm*, automates the process of navigation by suggesting to the user the best trail to follow, with respect to a query, given that the user is currently browsing a Web page. More specifically, given a starting URL of a Web page and a query it will present the user with the trail having maximal suitability to the query given the parameters input to the algorithm; we call this trail the *best trail*. The algorithm can be easily refined so that the n, with $n \geq 1$, most suitable trails can be returned instead of just the best trail; in addition, the algorithm can compute the best trails for multiple starting points. The user-interface, i.e. how the trails are presented to the user, can take many forms of a type known to those skilled in the art and need not be described in detail herein.

As will be appreciated, a navigation engine according to the present invention will ideally be supplied as a support tool for browsing accessible through a Web browser or as a plug-in to a search engine. In general, the present invention is applicable in any hypertext system, such as an electronic book, for the purpose of navigation assistance.

In the following Section 1 we define the terminology used in this document. In Section 2 we describe the working of the algorithm used by the navigation engine, the data structures used in the algorithm and the operations on these data structures. In Section 3 we give the pseudo-code and flowchart of the best trail

algorithm. In Section 4 we give an illustrative example of the working of the best trail algorithm. In Section 5 we describe a specific embodiment of the algorithm in a hypertext system.

1 Terminology

We now define the basic terms used in this document.

1) Each *Web page* (or simply page) has an associated URL which acts as a unique identifier or address for the purpose of locating the page and retrieving it. We consider URL in the generic sense, where a hypertext system other than the Web will also have some form of unique identification of its pages.

2) A *link* is an ordered pair of nodes from an anchor node to a destination node. Since nodes represent Web pages and these are uniquely identified by URLs, we consider a link to be an ordered pair of URLs. The *out-links* from a Web page are the links embedded within this page. The collections of links embedded in the pages of a hypertext system form a directed graph which determines its topology.

3) A *trail* is a sequence of URLs which is consistent with the topology of the Web. That is, any two adjacent URLs in the sequence form a link, which is embedded in the Web page identified by the anchor URL.

4) A *query* provides the goal of a user's navigation session. It is normally specified by the user as a set of keywords, for instance in the manner a query is specified to a search engine.

5) The *score of a URL* with respect to a query is the relevance or weight of the page associated with the URL with respect to the query. (At times we refer to the score of a URL as the score of its associated page.) That is, the score of a URL with respect to a query indicates how closely the page associated with the URL matches the query. We assume that the scoring function returns a numeric value, and that URLs with higher scores are more relevant to the query. In this

embodiment we also assume that all URLs have a positive (i.e. greater than zero) score, which in the case of a non-relevant URL will be small. The scoring function must be consistent in the sense that, within a navigation session, all URLs are scored in the same manner.

6) The *score of a trail* with respect to a query is a function of the scores of the individual Web pages of the trail with respect to the query. As for the scores of URLs, scores of trails are numeric and positive, and trails with higher scores are more suitable to the query.

Four possible scoring functions for trails are:

(a) The average score of its URLs with respect to the query.

(b) The average score of its distinct URLs with respect to the query (i.e. for the purpose of scoring the trail, each URL in the trail is counted only once even if a URL is revisited during the trail).

(c) The sum of the scores of its distinct URLs divided by the length of the trail; this scoring function penalises the trail when a URL is visited more than once.

(d) The sum of the discounted scores of its URLs with respect to the query, where the discounted score of U_i, the URL in the ith position in the trail, is the score of U_i with respect to the query multiplied by γ raised to the power of $(i - 1)$, where γ is a real number strictly between zero and one. I.e. the discounted score of a trail, $U_1, U_2, \ldots, U_m$, is equal to $\sum_{i=1}^{m} s_i \cdot \gamma^{i-1}$, where s_i is the score of U_i with respect to the query.

We can also combine scoring functions (c) and (d) by discounting in (c) each URL according to its previous number of occurrences within a trail.

7) An *ordering of trails* with respect to a query Q is defined as follows. Given two trails, T_1 and T_2, we say that T_1 is *better than* T_2 with respect to Q if the score of T_1 with respect to Q is greater or equal to the score of T_2 with respect to Q.

8) Let $\{T_1, T_2, \ldots, T_n\}$, $n \geq 1$, be a set of trails and Q be a query. The *rank* of a trail T_i in the set of trails, with respect to Q, is determined as follows. The trail with the highest score within the set is given the highest rank, i.e. 1, the trail with the second highest score within the set is ranked as 2, ..., and the trail with the lowest score within the set is given the lowest rank. Two trails with the same score are given the same rank, and all trail scores are with respect to Q.

9) *Browsing* is the general activity of exploring Web pages and inspecting their contents, and *navigation* is the activity of following links (colloquially known as "surfing").

2 Description of the Best Trail Algorithm

We describe the algorithm assuming one URL as its starting point; if the algorithm is embodied in a hypertext system other than the Web, then we assume that a mechanism exists for unique identification of its pages similar to the URL concept. In general, the algorithm will take as input several starting points and compute the best trail for each one of them; see the pseudo-code of the algorithm given in Section 3 and the flowchart of the algorithm shown in Figure 1.

Starting from the initial URL, the algorithm follows links from anchor to destination according to the topology of the Web or the hypertext under consideration (i.e. when an out-link exists from the Web page identified by its URL, then it may be traversed by the algorithm).

The algorithm builds a *navigation tree* (see Figure 3) whose root node U_1 is labelled by the URL of the starting point. Each time a destination URL is chosen, a new node U_2, U_3 is added to the navigation tree and is labelled by the destination URL. Nodes that may be added to the navigation tree as a result of traversing a link that has not yet been followed from an existing node are called *tip nodes*. We also consider the special case when a link has been traversed to a destination URL, and the page U_2 associated with this URL has no out-links.

Nodes in the navigation tree which are labelled by such URLs are called *leaf nodes*, and are also considered to be tip nodes.

At any given stage of the running of the algorithm, each tip node of the current state of the navigation tree is considered to be a destination node of an anchor of a link to be followed; in the case when the tip node is a leaf node, we can consider the destination node to be the leaf itself. The algorithm uses a random device to choose a tip node to be added to the navigation tree; in the special case when the tip node is a leaf node, the navigation tree remains unchanged. The weight that is attached to a tip node for the purpose of the probabilistic choice is proportional to the score of the trail induced by the tip node, which is the unique sequence of URLs labelling the nodes in the navigation tree forming a path from the root node of the tree to the tip node under consideration. (The exact formula for calculating the probability of a tip node is given in Section 3 as the value returned by the auxiliary function *prob*; see Equation 1.) We call the process of adding a tip node to the navigation tree *node extension*. The best trail algorithm terminates after a prescribed number of node extensions, each such extension being a single iteration within the algorithm.

The algorithm has two separate stages, the first being the *exploration stage* and the second being the *convergence stage*. Each stage comprises a preset number of iterations. During the exploration stage a tip node is chosen with probability purely proportional to the score of the trail that it induces. During the convergence stage we apply a "cooling schedule", where tip nodes which induce trails having higher scores are given exponentially higher weights at each iteration according to the rank of their trails, as determined by their trail scores, and the number of iterations completed so far in the convergence stage. A parameter called the *discrimination factor (df)*, which is a real number strictly between zero and one, determines the convergence rate. When the algorithm terminates the *best trail* is returned, which is the highest ranking trail induced by the tip nodes of the navigation tree. Convergence to the absolute best trail can be achieved provided the number of iterations in both stages of the algorithm is large enough and the discrimination factor is not too low. The best trail algorithm can be

modified so that the discrimination factor decreases dynamically during the convergence stage.

We now define the terms used in the algorithm.

1) A *navigation tree* is a tree whose *root* is the starting URL of a navigation session. The nodes in the navigation tree are labelled by URLs, where it is possible for two different nodes in the tree to be labelled by the same URL. Each arc in a navigation tree from one node (the anchor node) to another node (the destination node) corresponds to an existing link in the Web from the URL labelling the anchor node to the URL labelling the destination node.

2) A *frontier* node in a navigation tree is a node in this navigation tree that is either

(a) a *leaf* node, when the page associated with its URL has no out-links, or

(b) a node such that the page associated with its URL has one or more out-links to destination URLs that are *not* already labels of destination nodes of this frontier node. That is, the page of the URL associated with such a frontier node is the anchor node of a link that has not yet been traversed from this node.

We may assume that the destination URL of an out-link from the Web page associated with a frontier node is not the same as the URL labelling this frontier node, i.e. we would normally ignore cycles of length one which are present in the Web topology.

3) A *tip* node in a navigation tree is either

(a) a frontier node which is a leaf node, or

(b) a new node that is *not* already in the navigation tree such that the URL labelling this new node will be the destination node of an arc whose anchor is a frontier node. Moreover, there is *no* arc outgoing from this frontier node whose destination node is already labelled by the same URL as that of the new node. (So, the URLs labelling the destination node of a common anchor node are all distinct.) Such a frontier node is called the *parent* node of this tip node.

That is, the URL associated with such a tip node is the destination of a link that is embedded in the page associated with the URL of its parent frontier node and this link has not yet been traversed from this frontier node.

4) The *trail induced by a tip node* in a navigation tree is the unique sequence of URLs labelling the nodes in the navigation tree which form a path from the root node of the tree to the tip node under consideration.

The *score of the trail induced by a tip node* in a navigation tree, with respect to a query, is the score of the trail induced by the tip node with respect to the query.

5) The *extension* of a navigation tree with one of its tip nodes, which we call *node extension*, is done according to the following two cases:

(a) if the tip node is a leaf node, the navigation tree remains unchanged, otherwise

(b) add a new node and arc to the navigation tree such that the anchor node of this arc is the parent frontier node of this tip node and the destination node is the tip node itself. The new node becomes a frontier node of the extended navigation tree.

6) The *probability of a tip* in a navigation tree is given in Section 3 as the value returned by auxiliary function *prob*; see Equation 1 for the formula defining this value. This probability is proportional to the score of the trail induced by the tip. During the convergence stage of the algorithm the probability is exponentially higher for trails having higher rank as determined by the discrimination factor.

3 Pseudo-Code and Flowchart of the Best Trail Algorithm

<u>Inputs</u>:

1) A query Q.
2) An indexed set $\{U_1, U_2, \ldots, U_N\}$ of N URLs, where $N \geq 1$.

3) A positive integer, $M \geq 1$, which specifies the number of repetitions of the algorithm for each input URL.

Output:

An indexed set $\{B_1, B_2, \ldots, B_N\}$ consisting of N trails, one for each input URL; if required this set of trails may be *ranked* such that B_1 is better than B_2 with respect to Q, B_2 is better than B_3 with respect to Q, ..., and B_{N-1} is better than B_N with respect to Q.

Global parameters:

1) df, where $0 < df < 1$, is called the *discrimination factor.*

2) $I_{explore} \geq 0$ is the number of iterations during the exploration stage of the algorithm.

3) $I_{converge} \geq 1$ is the number of iterations during the convergence stage of the algorithm.

4) $\{D_1, D_2, \ldots, D_M\}$ is an indexed set of M navigation trees for each input URL U_k, $1 \leq k \leq N$. Initially $D_i = \{U_k\}$, i.e. D_i is a navigation tree having a single node U_k, which is also its root, where $1 \leq k \leq N$.

Definitions of auxiliary functions:

1) *extend* (D_i, t), where D_i is a navigation tree and t is a tip of D_i, returns a navigation tree resulting from the extension of D_i with t.

2) *score* (Q, D_i, t), where Q is a query, D_i is a navigation tree and t is a tip of D_i, returns the score of the trail induced by t with respect to Q.

3) *rank* (Q, D_i, t), where Q is a query, D_i is a navigation tree and t is a tip of D_i, returns the rank of the trail induced by t with respect to Q, within the set of trails induced by the tip nodes of D_i.

4) *prob* (Q, D_i, t, x, j), where Q is a query, D_i is a navigation tree, t is a tip of D_i, x is either 1 or *df*, and j is a positive integer, denoting the exploration or convergence step, returns

$$prob(Q,D_i,t,x,j) = \frac{score(Q,D_i,t) \cdot power(x, rank(Q,D_i,t) \cdot j)}{\sum_{k=1}^{n} score(Q,D_i,t_k) \cdot power(x, rank(Q,D_i,t_k) \cdot j)} \quad (1)$$

where $\{t_1, t_2, \ldots, t_n\}$ is the set of tip nodes of D_i, *power* (x, y) is a shorthand for x raised to the power of y and $x \cdot y$ is a shorthand for the multiplication of x and y.

The interpretation of *prob*(Q, D_i, x, j) is the probability of a tip t in the navigation tree D_i, with respect to the query Q.

5) *select*(Q, D_i, x, j), where Q is a query, D_i is a navigation tree, x is either 1 or *df*, and j is a positive integer, returns a tip of D_i chosen by a random device operating according to the probability distribution function *prob*(Q, D_i, t, x, j).

6) *best*(Q, D_i), where Q is a query and D_i is a navigation tree, returns the trail with the highest score from the set of trails induced by the set of tip nodes of D_i.

7) *overall_best*(Q, $\{T_1, T_2, \ldots, T_M\}$), where Q is a query and $\{T_1, T_2, \ldots, T_M\}$ is a set of M trails, returns the highest scoring trail from this set. We call this trail the *best trail*, since it is the highest ranking trail traversed by the algorithm, given a starting URL, U_k, and an input query, Q.

<u>Pseudo-code of the algorithm</u>:

This is given as Algorithm 1; the flowchart of the algorithm is given in Figure 1. The algorithm has a main outer for loop starting at line 2 and ending at line 16, which computes the best trail for each one of the N input URLs. The first inner for loop starting at line 3 and ending at line 14 recomputes the best trail M times,

given the starting URL U_k. The overall best trail out of the M iterations with the same starting URL, is chosen at line 15 of the algorithm. We note that due to the stochastic nature of the algorithm, we may get different trails T_i at line 13 of the algorithm from two separate iterations of the for loop starting at line 3 and ending at line 14. The algorithm has two further inner for loops, the first one starting at line 5 and ending at line 8 comprises the exploration stage of the algorithm, and the second one starting at line 9 and ending at line 12 comprises the convergence stage of the algorithm. Finally, the set of N best trails for the set of N input URLs is returned at line 17 of the algorithm.

```
Algorithm 1: Best_Trail(Q,{U1,U2,...,UN},M)
1.  begin
2.      for k = 1 to N do
3.          for i = 1 to M do
4.              Di ← {Uk};
5.              for j = 1 to I_explore do
6.                  t ← select(Q,Di,1,j);
7.                  Di ← extend(Di, t);
8.              end for
9.              for j = 1 to I_converge do
10.                 t ← select(Q, Di, df, j);
11.                 Di ← extend(Di, t);
12.             end for
13.             Ti ← best(Q, Di);
14.         end for
15.         Bk ← overall_best (Q,{T1,T2,...,TM});
16.     end for
17.     return {B1, B2, ..., BN}:
18. end.
```

4 Example of the Working of the Algorithm

In Figure 2 we show an example Web topology, where each node is annotated with its URL and the score of this URL with respect to a given query is given in parentheses. Assuming that U_1 is the starting URL of the best trail algorithm, a possible navigation tree after seven node extensions is given in Figure 3. Each node in the navigation tree is annotated with a unique number and with its URL; the tip nodes of the navigation tree are shaded. The root of the navigation tree is node 0, which is labelled by the starting URL U_1, and the nodes that were added to the navigation tree as a result of the seven node extensions are numbered from 1 to 7. Dashed nodes and arcs indicate URLs and links that were, respectively, previously visited and traversed.

The frontier nodes of the navigation tree are 1, 5, 6 and 7. Node 1 is also a tip node of the navigation tree since it is a leaf node. Node 5 is the parent of two tip nodes, numbered 8 and 9. Node 6 is the parent of one tip node, numbered 10. Similarly, node 7 is the parent of two tip nodes, numbered 11 and 12. Table 1 shows the tips, their induced trails and the score of these trails according to the first three trail scoring functions suggested in Section 1 term (6). (As will be noted, in this example, the trail to tip 11 is considered the best trail (i.e. highest score) irrespective of the scoring function (a,b,c) used.) Using this table the probability of the next tip node to add to the navigation tree can be computed. As can be seen these probabilities are, in general, different for different trail scoring functions.

TIP	INDUCED TRAIL	SCORE (a)	SCORE (b)	SCORE (c)
1	U_1,U_2	2.00	2.00	2.00
8	U_1,U_3,U_4,U_1,U_2	2.40	2.75	2.20
9	U_1,U_3,U_4,U_1,U_3	2.20	2.66	1.60
10	U_1,U_3,U_5,U_6,U_1	2.60	3.00	2.40
11	U_1,U_3,U_5,U_6,U_3,U_4	3.17	3.40	2.83
12	U_1,U_3,U_5,U_6,U_3,U_5	2.83	3.00	2.00

Table 1: The trails induced by the tips and their scores

5 Industrial Application

We see the main application of the best trail algorithm as a support tool for browsing or as a plug-in to a search engine in order to assist users during navigation. In general, the best trail algorithm is applicable in any hypertext system, such as an electronic book, for the purpose of navigation assistance.

For example, as a plug-in to a search engine the algorithm could be used for the purpose of calculating and displaying to the user the best trail for each of the top URLs that match the input query. As a navigation support tool for browsing, the user would be asked to input using a keyboard 20 or mouse 22 (for example) a query and using the destination URLs of the links embedded in the currently browsed Web page as the starting points for navigation, the browser would display on a screen 24 to the user the best trail for each one of these URLs. The algorithm can be easily refined so that for each starting URL the n, with $n \geq 1$, most relevant trails can be returned rather than just the best trail. This process can be repeated after the user follows a link.

As should be appreciated, a navigation engine and system according to the present invention provide very useful tools for use when navigating within a hypertext system, such as the World-Wide-Web. Further, although a complete software listing has not been provided herein, it will be immediately obvious to a person skilled in the relevant art as to how to put this invention into practice once the algorithm described herein is known. Hence, it is considered that the specification of this patent application is fully sufficient to support the invention as claimed.

It will of course be understood that the present invention has been described above purely by way of example, and that modifications of detail can be made within the scope of the appended claims.

CLAIMS

1. A navigation engine which uses a query defining a subject of interest to a user to select links between relevant pages in a network of linked textual or multi-media information, the navigation engine being able to assess the suitability of a plurality of links forming a trail based on the relevance of the pages in the trail, wherein the navigation engine provides an output related to the suitability to a user of various trails assessed.

2. A navigation engine as claimed in claim 1, wherein the output includes a list of suitable trails available to be accessed by a user.

3. A navigation engine as claimed in claim 2, wherein the list of suitable trails is in order of suitability, with the most suitable trail listed first.

4. A navigation engine as claimed in any preceding claim, wherein relevance of a page in a network is assessed based on the relevance of the page with respect to a query.

5. A navigation engine as claimed in any preceding claim, wherein a score is allocated to indicate the relevance of a page with respect to a query.

6. A navigation engine as claimed in any preceding claim, wherein suitability of a trail is calculated based on a chosen scoring function.

7. A navigation engine as claimed in claim 6, wherein the scoring function involves at least one of the following:

(a) the average score for a page in the trail with respect to a query, taking into account each step in the trail;

(b) the average score of a page in the trail with respect to a query, counting each page only once even if it appears in the trail more than once;

(c) the sum of the scores of pages in the trail with respect to a query, counting each distinct page only once even it appears more than once in the trail,

divided by the total number pages in the trail irrespective of whether a page appears more than once in the trail; and

(d) the sum of discounted scores of the pages in the trail with respect to a query, were the discounted score of U_i, the page in the ith position in the trail, is the score of U_i with respect to the query multiplied by γ raised to the power of $(i - 1)$ where γ is a real number strictly between 0 and 1, i.e. the discounted score of a trail, $U_1, U_2, \ldots, U_m$, is equal to $\sum_{i=1}^{m} s_i \cdot \gamma^{i-1}$, where s_i is the score of U_i with respect to the query.

8. A navigation engine as claimed in claim 6 or claim 7, wherein the trails are ordered based on the result of the scoring function.

9. A navigation engine as claimed in any preceding claim, wherein the trails are ranked by score, the highest score reflecting the best trail.

10. A navigation engine as claimed in any preceding claim, wherein the trails can end with pages having no out-link.

11. A navigation engine as claimed in any preceding claim, wherein assessment of a plurality of trails comprises an exploration stage and a convergence stage.

12. A navigation engine as claimed in claim 11, wherein the exploration stage includes extending trail lengths and scoring the trails that are induced.

13. A navigation engine as claimed in claim 11 or claim 12, wherein the convergence stage assesses which induced trails are more suitable and gives these trails more weight at each iteration based on their ranking.

14. A navigation engine as claimed in any preceding claim, wherein an assessment of trails is conducted over sufficient iterations to produce a useful output.

15. A navigation engine as claimed in any preceding claim, wherein the network is a hypertext system, such as the World-Wide-Web.

16. A navigation engine as claimed in any preceding claim which can be loaded into a computer system for connection to a network.

17. A navigation engine which uses a query or queries defining a subject of interest to a user to select links between relevant pages in a network, substantially as hereinbefore described with reference to and as shown in the accompanying drawings.

18. A system for facilitating exploration by a user of a network of linked textual or multi-media information, the system comprising:

a user interface for receiving a query which defines a subject of interest to the user; and

a navigation engine as claimed in any preceding claim.

19. A system for facilitating exploration by a user of a network of linked textual or multi-media information, substantially as hereinbefore described with reference to and as shown in the accompanying drawing.

20.

Application No:	GB 9930070.9	**Examiner:**	Geoff Western
Claims searched:	1-19	**Date of search:**	4 August 2000

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.R): G4A (AUDB)

Int Cl (Ed.7): G06F 17/30

Other: Online : COMPUTER, EPODOC, INSPEC, Internet, JAPIO, TDB, WPI

Documents considered to be relevant:

Category	Identity of document and relevant passage	Relevant to claims
A,E	EP 0981097 A1 (SOLAR INFORMATION)	-
X	"Toward user-centric navigation of the web: COOL links using SPI", as last modified 25 April 1997, at www.scope.gmd.de/info/www6/posters/744/ucn.htm	-

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
		E	Patent document published on or after, but with priority date earlier than, the filing date of this application.
&	Member of the same patent family		